Task 4

The problem that still persists even after deriving a cpa secure encryption scheme is that while cpa guarantees confidentiality, it cannot tell anything about integrity. So, while we know that the message sent by Alice to Bob can't be read by anyone else, however, someone could intercept the message change the cipher text and then forward it to bob. This can be particularly dangerous when the integrity of the message is essential for example banking applications or military applications.

The aim of message authentication code therefore is to prevent an adversary from modifying a message sent by one party to another, without the parties ever detecting a modification has been made.

A MAC is therefore an algorithm that is applied to the message. The output of the algorithm is a MAC tag that is sent along the message. Security is formulated by requiring that no adversary can generate a valid MAC tag on any message that was not sent by the legitimate parties.

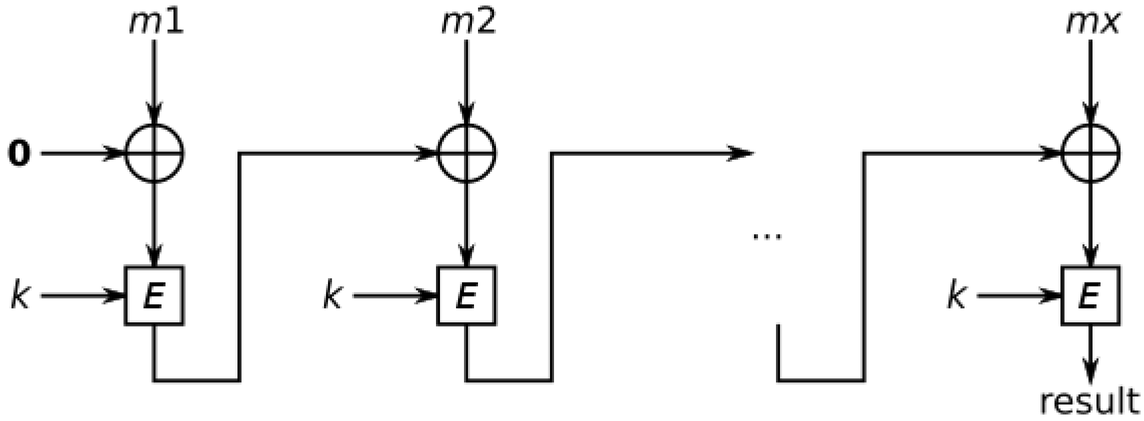
Build Algorithm

A MAC algorithm is built from three algorithms, Gen, Mac and Vrfy. Gen generates a secret key, MAC generates MAC tags. Vrfy receives a key, k, a message m, a tag t and outputs either 1 or 0 meaning the MAC tag is valid or invalid respectively.

CBC-MAC Algorithm

The issue with the generic MAC algorithm is that in order to calculate MAC on a message of length l.n, the mac tag generated is of length 4l.n which is a serious issue especially if the messages to be sent are large.

CBC Mac divides up the message similar to how we did in randomized counter mode, the block cipher is applied l times on a message of length l.n, more importantly the output tag is of size n bits.



Basic MAC:

1) Tag t is set to 0, and message block 1 is xor with the tag, then encryption function E is applied as created in task2.
2) The output of each block is fed to next block and the result is our final Tag.

However, this construction is only secure when the length of messages is fixed and for variable length messages, modifications are needed.

The variation of MAC I have used is:

Two different keys, k1 and k2 are chosen. Then, compute the basic MAC using key k1, on the result t1, we again compute function F(t1) with key k2. The result is the final tag.

Proof

Let Q be the set of all queries from adversary to oracle.

To define the security of MAC, we define a game such that output of the game is 1 if and only if

$$Vrfy(m, t) = 1, m \text{ is not in } Q$$

A MAC is secure if Pr[MAC_Game(n) == 1] <= negl(n)

If F is a PRF, then CBC is a PRF as long as the set of inputs on which it is queried is prefix-free. Formally, for all probabilistic polynomial time (p.p.t.) distinguishers D that query their oracle on a prefix-free set of inputs, there is a negligible function negl() such that

$$|Pr[DCBCk(.)\ (1n\ ) = 1] - |P\ r[Df(.)\ (1n\ ) = 1] \leq negl(n)\ (1)$$